

SUBROUTINE TO SCROLL SCREEN DATA 1 PIXEL RIGHT

<u>BYTE #</u>	<u>DECIMAL</u>	<u>HEX</u>	<u>MNEMONICS</u>	<u>LABELS</u>	<u>COMMENTS</u>
0	213	D5	PUSH DE	START	Save BASIC pointer.
1	175	AF	XOR A		Clear the A register.
2	33	21	LD HL, nn		Set pointer to top of
3	0	00			screen RAM.
4	64	40	DW 16384 ₁₀		
5	14	0E	LD C, n		Set # lines to scroll.
6	90	5A	DB 90 ₁₀		
7	6	06	LD B, n	LOOP 2	Set # bytes per line
8	40	28	DB 40 ₁₀		to scroll.
9	203	CB	RES 0, D	LOOP 1	Clear bit 0 of D register.
10	130	82			
11	126	7E	LD A, (HL)		Get a byte.
12	230	E6	AND n		Mask out program data.
13	170	AA	DB 170 ₁₀		
14	31	1F	RRA		Scroll area to right 2
15	203	CB	RRD		bits (1 pixel) and
16	26	1A			move LSB to MSB thru
17	31	1F	RRA		Carry Flag and D register.
18	203	CB	RLD		
19	18	12			
20	174	AE	XOR (HL)		Mask program out of
21	230	E6	AND n		shifted byte;
22	170	AA	DB, 170 ₁₀		restore screen and
23	174	AE	XOR (HL)		program mixture.
24	119	77	LD (HL), A		Scrolled byte to screen.
25	35	23	INC HL		Next byte address.
26	16	10	DJNZ, dis		Decrement B; if not 0,
27	239	EF	DB -17 ₁₀		jump to LOOP 1; else
28	13	0D	DEC C		Decrement C;
29	32	20	JRNZ, dis		if C#0, jump to
30	232	E8	DB -24 ₁₀		LOOP 2; else
31	209	D1	POP DE		Restore BASIC pointer.
32	201	C9	RET		Return to BASIC.

SUBROUTINE TO SCROLL SCREEN DATA 1 PIXEL LEFT

<u>BYTE #</u>	<u>DECIMAL</u>	<u>HEX</u>	<u>MNEMONICS</u>	<u>LABELS</u>	<u>COMMENTS</u>
0	213	D5	PUSH DE	START	Save BASIC pointer.
1	175	AF	XOR A		Clear the A register.
2	33	21	LD HL, nn		Set pointer to bottom
3	15	0F			of screen RAM.
4	78	4E	DW 19983 ₁₀		
5	14	0E	LD C, n		Set # lines to scroll.
6	90	5A	DB 90 ₁₀		
7	6	06	LD B, n	LOOP 2	Set # bytes per line
8	40	28	DB 40 ₁₀		to scroll.
9	203	CB	RES 0, D	LOOP 1	Clear bit 0 of D register.
10	130	82			
11	126	7E	LD A, (HL)		Get a byte.
12	230	E6	AND n		Mask out program data.
13	170	AA	DB 170 ₁₀		
14	31	1F	RRA		Scroll area to right 2
15	203	CB	RRD		bits (1 pixel) and
16	26	1A			move LSB to MSB thru
17	31	1F	RRA		Carry Flag and D register.
18	203	CB	RLD		
19	18	12			
20	31	1F	RRA		Scroll area to right 2
21	203	CB	RRD		bits (1 pixel) and
22	26	1A			move LSB to MSB thru
23	31	1F	RRA		Carry Flag and D register.
24	203	CB	RLD		
25	18	12			
26	31	1F	RRA		Scroll area to right 2
27	203	CB	RRD		bits (1 pixel) and
28	26	1A			move LSB to MSB thru
29	31	1F	RRA		Carry Flag and D register.
30	203	CB	RLD		
31	18	12			
32	31	1F	RRA		Scroll area to right 2
33	203	CB	RRD		bits (1 pixel) and
34	26	1A			move LSB to MSB thru
35	31	1F	RRA		Carry Flag and D register.
36	203	CB	RLD		
37	18	12			
38	174	AE	XOR (HL)		Mask program out of

left

39	230	E6	AND n	shifted byte;
40	170	AA	DB, 170 ₁₀	restore screen and
41	174	AE	XOR (HL)	program mixture.
42	119	77	LD (HL), A	Scrolled byte to screen.
43	35-43	23 2B	DEC INC HL	Next byte address.
44	16	10	DJNZ, dis	Decrement B; if not 0,
45	221	00	DB -35 ₁₀	jump to LOOP 1; else
46	13	00	DEC C	Decrement C;
47	32	20	JRNZ, dis	if C#0, jump to
48	214	06	DB -42 ₁₀	LOOP 2; else
49	209	01	POP DE	Restore BASIC pointer.
50	201	C9	RET	Return to BASIC.

26 September 1982

PSC Box 542
APO Miami 34004

5-1
corrected

Bob Fabris
2636 Morrie Drive
San Jose, CA 95127

Dear Bob,

In answer to the challenge on page 105 of the Arcadian for scrolling the screen up, down, left and right, enclosed are:

- A disassembled listing of the "scroll right" routine on page 105;
- A listing to scroll the screen 1 pixel left;
- A listing to scroll the screen 1 pixel up;
- A listing to scroll the screen 1 pixel down.

Listing "D" requires that either the top line (Y=43) be blank, or a BOX 0,43,160,1,2 command be issued before calling the machine-language subroutine - otherwise, anything in the foreground color on line 43 will be "stretched" down each time the routine is called. This is due to the fact that there is no RAM "above" line 43 - by changing the byte at location 9 to 90₁₀, you can cause a curtain effect which covers each line in the foreground color.

SUBROUTINE TO SCROLL SCREEN DATA 1 PIXEL UP

<u>BYTE #</u>	<u>DECIMAL</u>	<u>HEX</u>	<u>MNEMONICS</u>	<u>TAGS</u>	<u>COMMENTS</u>
0	213	D5	PUSH DE	START	Save BASIC pointer.
1	175	AF	XOR A		Clear the A register.
2	33	21	LD HL, nn		Set pointer to top of
3	0	00			screen RAM.
4	64	40	DW 16384 ₁₀		
5	17	11	LD DE, nn		Address of next byte up.
6	40	28			
7	64	40	DW 16424 ₁₀		
8	14	0E	LD C, n		Set # lines to scroll.
9	90	5A	DB 90 ₁₀		
10	6	06	LD B, n	LOOP 2	Set # bytes per line
11	40	28	DB 40 ₁₀		to scroll.
12	126	7E	LD A, (HL)	LOOP 1	Get a byte.
13	230	E6	AND n		Mask out screen data.
14	85	55	DB 85 ₁₀		
15	119	77	LD (HL), A		Save program data.
16	26	1A	LD A, (DE)		Get next byte.
17	230	E6	AND n		Mask out program data.
18	170	AA	DB 170 ₁₀		
19	174	AE	XOR (HL)		Move up one line and mix
20	119	77	LD (HL), A		with program there.
21	19	13	INC DE		Next pair of
22	35	23	INC HL		bytes' addresses.
23	16	10	DJNZ, dis		Decrement B; if not 0,
24	243	F3	DB -13 ₁₀		jump to LOOP 1; else
25	13	0D	DEC C		Decrement C;
26	32	20	JRNZ, dis		if C#0, jump to
27	238	EE	DB -18 ₁₀		LOOP 2; else
28	209	D1	POP DE		Restore BASIC pointer.
29	201	C9	RET		Return to BASIC.

SUBROUTINE TO SCROLL SCREEN DATA 1 PIXEL DOWN

<u>BYTE #</u>	<u>DECIMAL</u>	<u>HEX</u>	<u>MNEMONICS</u>	<u>LABELS</u>	<u>COMMENTS</u>
0	213	D5	PUSH DE	START	Save BASIC pointer.
1	175	AF	XOR A		Clear the A register.
2	33	21	LD HL, nn		Set pointer to bottom of screen RAM.
3	15	0F			
4	78	4E	DW 19983 ₁₀		
5	17	11	LD DE, nn		Address of next byte up.
6	231	E7			
7	77	4D	DW 19943 ₁₀		
8	14	0E	LD C, n		Set # lines to scroll.
9	89	59	DB 85 ₁₀		
10	6	06	LD B, n	LOOP 2	Set # bytes per line to scroll.
11	40	28	DB 40 ₁₀		
12	126	7E	LD A, (HL)	LOOP 1	Get a byte.
13	230	E6	AND n		Mask out screen data.
14	85	55	DB 85 ₁₀		
15	119	77	LD (HL), A		Save program data.
16	26	1A	LD A, (DE)		Get next byte.
17	230	E6	AND n		Mask out program data.
18	170	AA	DB 170 ₁₀		
19	174	AE	XOR (HL)		Move down one line and mix with program there.
20	119	77	LD (HL), A		
21	27	1B	DEC DE		Next pair of bytes' addresses.
22	43	2B	DEC HL		
23	16	10	DJNZ, dis		Decrement B; if not 0, jump to LOOP 1; else
24	243	F3	DB -13 ₁₀		
25	13	0D	DEC C		Decrement C;
26	32	20	JRNZ, dis		if C#0, jump to
27	238	EE	DB -18 ₁₀		LOOP 2; else
28	209	D1	POP DE		Restore BASIC pointer.
29	201	C9	RET		Return to BASIC.